OFTWARE SOFTWARE SOFT

SISTEM DE OPERARE PE DISC FLEXIBIL

PREZENTARE

INSTITUTUL DE CERCETARE ŞTIINŢIFICĂ ŞI INGINERIE
TEHNOLOGICĂ PENTRU TEHNICA DE CALCUL ŞI INFORMATICĂ
SECTOR TEHNICĂ DE CALCUL

COBRA

SISTEM DE OPERARE PE DISC FLEXIBIL

BRAŞOV
1988

COORDONATORII SERIEI
dr. ing. Dan Roman
dr. Emil Munteanu

TEHNOREDACTORI

COPERTA SERIEI
designer Liviu Derveşteanu

Implementarea sistemului de operare CP/M s-a realizat de un colectiv de la I.T.C.I. filiala Braşov format din: mat. Arefta Marcel, mat. Pop Mircea condus de: dr. ing. Toacse Gheorghe

CUPRINS

INTRODUCERE

Microsistemul COBRA este un calculator care la dorinţa utilizatorului se poate configura în maşina BASIC-SPECTRUM sau CP/M.
La pornire, pe ecranul televizorului, se afişează emblema microsistemului şi se aşteaptă introducerea unei opţiuni.

Opţiuni:

1. Încărcarea sistemului de operare CP/M:
   — prin apăsarea tastei „D"
2. Încărcarea interpretorului BASIC-SPECTRUM:
   — de pe caseta magnetică prin apăsarea tastei „C"
   — din memoria EPROM prin apăsarea tastei „B"

Observaţii:

— încărcarea sistemului de operare CP/M presupune existenţa pe disc a fişierului SYS.COM

— nu mai este necesar utilitarul SYSGEN, deoarece încărcarea sistemului de operare se face din fişier, deci multiplicarea acestuia se poate face cu unul din utilitarele DIP sau PIP

Precizări suplimentare privind organizarea şi funcţionarea sistemului în cele două regimuri se pot obţine consultînd manualele: COBRA BASIC, COBRA CP/M.

Acest manual descrie organizarea sistemului CP/M (inclusiv organizarea memoriei) şi punctele de intrare în sistem. Se vor prezenta informaţiile necesare pentru scrierea de programe executabile sub CP/M, programe ce utilizează facilităţile de I/E şi de lucru cu discul oferite de sistem.

1.1 ORGANIZAREA CP/M

Sistemul CP/M este alcătuit din punct de vedere logic din următoarele patru părţi:

BIOS — sistemul de I/E de bază, care oferă interfaţa cu perifericele

BDOS — sistemul de exploatare a discurilor, care oferă primitivele de acces la disc

CCP — procesorul de comenzi-consola

TPA — zona pentru programe tranzitorii.

Componentele BIOS şi BDOS sînt grupate într-un singur program numit FDOS, care are un punct de intrare unic. Componenta CCP este un program distinct, care utilizează programul FDOS pentru a oferi o interfaţă flexibilă între utilizator şi informaţiile existente pe disc. TPA este o zonă de memorie (i.e. zona de memorie care nu este utilizată de FDOS şi CCP) în care se execută comenzile tranzitorii CP/M şi programele-utilizator de aplicaţii. **Organizarea memoriei într-un sistem standard CP/PM este:**

0000 BOOT:	
	parametri sistem
TBASE:	T P A
CBASE:	C C P
FBASE:	F D O S (BDOS+BIOS)
	FFFF

OBSERVAŢIE: De obicei adresa BOOT este egală cu 0000H, adresa TPA este egală cu BOOT+0100H=0100H, iar adresele CBASE şi FBASE depind de tipul sistemului CP/M.

Adresele 0000H — 0007H sînt rezervate pentru:

0000H — 0002H	salt la rutina de reiniţializare a sistemului CP/M existentă în BIOS (JMP WBOOT)
0003H	octetul IOBYTE
0004H	numărul utilizatorului curent şi al discului instalat
0005H — 0007H	salt la punctul de intrare în FDOS, respectiv în BDOS (JMP FBASE)
0008H — 003FH	adresa de salt pentru instrucţiunile „RST n" neutilizate de sistem. Programele de depanare dinamică DDT, SID utilizează pentru punctele de întrerupere, instrucţiunea RST 7.
0040H — 005BH	neutilizaţi de sistem.
005BH — 007CH	blocul de control fişier implicit pregătit de **CCP** pentru programele încărcate în **TPA**.
007DH — 00FFH	zona tampon de I/O implicită pentru operaţii cu discul (conţine sectorul citit sau scris). zona utilizată de **CCP** pentru a transmite programelor încărcate linia de comandă zona utilizată ca stiva de lucru pentru programele executate în **TPA**.

OBSERVAŢII:

a. Adresa 0005H este **PUNCT DE INTRARE** din programe tranzitorii în rutinele sistemului CP/M (în BDOS)

b. Adresa prezentă în locaţiile 0006H — 0007H poate fi folosită pentru a determina dimensiunea maximă a memoriei disponibile (presupunînd că se reacoperă componenta CCP)

c. Adresa 0003H este rezervată pentru octetul IOBYTE (configuraţia de I/E curentă)

d. Adresa 0004H este rezervată pentru a stoca numărul utilizatorului curent şi numărul discului instalat; octetul de la această adresă are forma:

număr utilizator curent	**număr disc instalat**
0000 — 1111	**0000 — 1111** **A—P**

1.1.1. PARTICULARITĂŢI ALE SISTEMULUI CP/M — COBRA

Sistemul **CP/M** implementat pe microcalculatorul **COBRA** prezintă unele diferenţe faţă de sistemul standard **CP/M 2.2** elaborat de firma **DIGITAL RESEARCH**, dar este compatibil cu acesta în ceea ce priveşte formatul volumelor şi fişierelor.

1.1.2. UTILIZAREA MEMORIEI INTERNE

Harta memoriei interne a sistemului **COBRA:**

Adresă	Zonă
FFFFH	B I O S
FA00H	B D O S
EC00H	C C P
E400H	B I O S
DE00H	RAM VIDEO
C000H	GENERATOR CARACTERE B U F F E R E
A800H	T P A
0100H	ZONA SISTEM
000H	

1.1.3. ORGANIZAREA DISCULUI

Sistemul de operare **CP/M** implementat pe microcalculatorul **COBRA** recunoaşte următoarele formate de disc:

— format standard 8" simplă densitate (compatibil **M18, M118, JUNIOR, CUBZ,...**)

— format standard 5" 1/4 dublă densitate 512 octeţi/sector (compatibil **JUNIOR,...**)

— format COBRA — 8" simplă densitate (directory în pista 0)
— 5" 1/4 dublă densitate (directory în pista 0)

Caracteristici:

— disc flexibil 8":

— 77 piste/disc
— 26 sectoare/pistă
— 128 octeţi/sector

— disc flexibil 5" 1/4

— dublă densitate (format **IBM**)

— 40 piste/disc
— 9 sectoare/pistă
— 512 octeţi/sector

— dublă densitate (format COBRA)

— 40 piste/disc
— 10 sectoare/pistă
— 512 octeţi/sector

Alegerea unuia din formatele de mai sus se face folosind utilitarul **CDISK** prezentat în „Manual de utilizare" al microcalculatorului **COBRA.**

1.1.4 DISPOZITIVUL CONSOLĂ

O altă particularitate a microcalculatorului **COBRA** este faptul că el nu foloseşte ca dispozitiv consolă un terminal care să realizeze funcţiile de intrare, ieşire şi de editare, aşa cum este cazul celorlalte microcalculatoare din aceeaşi categorie, care au implementate sistemul de operare **CP/M.** Toate funcţiile de terminal amintite mai sus sînt realizate de un set de rutine specializate (dependente de maşină), implementate chiar în componenta **BIOS** a **CP/M.** Dăm în continuare o descriere a facilităţilor hard şi a soluţiei soft adoptate, în vederea utilizării eficiente a unor elemente mai simple şi mai ieftine, în locul unui terminal specializat: tastatura matricială **QWERTY** (în intrare), **TV** sau monitor **TV** alb-negru sau color (în ieşire).

1.1.4.1 *Tastatura*

Este utilizată o tastatură tip **QWERTY** cu 6 linii ×8 coloane, deci 48 poziţii independente, tastabile. Parte din aceste taste sînt dublate, triplate prin tastarea simultană a două taste (de ex. tasta SHIFT sau CTRL şi încă o tastă), obţinîndu-se astfel codurile literelor mici, ale semnelor grafice speciale şi ale controalelor.

Deoarece spaţiul de afişare are dimensiunea de 24 linii × 32 coloane, el este considerat numai ca o fereastră ce se mişcă lateral (stînga sau dreapta) în cadrul unui spaţiu cu dimensiunea de 24 linii × 80 coloane, iar aceste deplasări pot fi comandate din tastatură. De asemenea mai poate fi comandată din tastatură şi viteza de execuţie a scrollului vertical şi numărul de coloane cu care se deplasează fereastra în bufferul alfanumeric mare. Viteza de execuţie a scrollului vertical este dată de faptul că acesta se realizează (opţional) pe 1, 2, 4 sau 8 linii **TV.** Iată aceste comenzi:

←	scroll stînga 1 coloana
→	scroll dreapta 1 coloana
GRAFICS	revenire în coloana 0
↓	la o tastare scade viteza cu o treaptă
↑	lă o tastare creşte viteza cu o treaptă
NOSCROLL	se întrerupe afişarea pînă la retastare NOSCROLL

1.1.4.2 *Afişajul*

Ca dispozitiv de afişare se foloseşte un aparat **TV** (obişnuit) alb-negru sau color. Procesul care realizează afişarea informaţiilor pe ecranul **TV** este

pe scurt următorul: microprocesorul Z80 şi componenta numită controller video, îşi dispută pe rînd accesul la o zonă de memorie numită **RAM VIDEO**, controllerul video explorînd în citire această zonă de memorie, afişează fiecare bit pe ecranul **TV** de exemplu în convenţia 0 punct stins, 1 punct aprins (aceasta la microcalculatoarele **PRAE**, **AMIC** sau **ZX80**).

Dimensiunile zonei afişate pe ecranul **TV** sînt de 192×256 puncte (biţi). Acest spaţiu se consideră a fi împărţit în matrici de 8×8 biti (puncte) deci în 24×32 matrici de 8×8 puncte. În continuarea acestei zone de memorie ce conţine informaţie afişabilă, urmează o zonă de memorie ai cărei octeţi se consideră în următoarea convenţie: fiecare octet din această zonă condiţionează biunivoc afişarea unei matrici de 8×8 puncte din zona descrisă mai sus. Această condiţionare se realizează prin următoarea structurare funcţională a biţilor din octeţii acestei de a doua zone, numite memorie cu atribute:

F	B	Ink	Paper
7	6	543	210

În această convenţie se depăşeşte partajarea făcută mai sus în 0 punct stins, 1 punct aprins, ajungîndu-se la următoarele: pe o matrice de 8×8 puncte se pot defini două culori, una constituind fondul (hîrtia) reprezentată de biţii de valoare 0 din matrice şi căreia i se atribuie culoare definită de biţii 0—2 din octetul atribut, cealaltă culoare constituind scrisul (cerneala) reprezentată de biţii de valoare 1 din matrice şi căreia i se atribuie culoarea definită de biţii 3—5 din octetul atribut. Bitul 6 dă posibilitatea definirii a două nuanţe pentru fiecare culoare şi se numeşte atribut de strălucire (0 normal, 1 cu strălucire), ridicînd deci la 16 numărul de culori. Cele opt culori selectabile se desemnează astfel:

000 **negru**
001 **albastru**
010 **roşu**
011 **magenta**
100 **verde**
101 **cyan**
110 **galben**
111 **alb**

Bitul 7 din octetul atribut validează funcţionarea blinking (schimbarea alternativă a culorii fondului cu aceea a scrisului).

Pe ecranul **TV**, zona grafică de 256× 192 pixeli este încadrată de o margine numită **BORDER** a cărei culoare poate fi definită de utilizator în convenţia de culoare de mai sus cu ajutorul funcţiei de terminal CTRL/B.

Modificarea culorii **PAPER**-ului şi a **INK**-ului, modificarea stării de strălucire şi blinking se face cu ajutorul funcţiei de terminal CTRL/C.

Iată în continuare harta zonelor de memorie descrise mai înainte:

adresa început —	adresa sfîrşit —	conţinut
C000H	C7FFH	prima treime
C800H	CFFFH	a doua treime
D000H	D7FFH	a treia treime
D800H	DAFFH	zona cu atribute

Memoria grafică constituită din cele 24 × 32 matrici de 8×8 biţi este organizată intreţesut în zona C000H — D7FFH adresarea făcîndu-se cu următoarea convenţie de adresare:

1	1	0	T1	T0	R2	R1	R0	L2	L1	L0	O4	O3	O2	O1	O0

unde R2 R1 R0 este o grupare de trei biţi ce reprezintă numărul rîndului dintr-o linie de 32 de matrici de 8×8 biţi de afişare alfanumerică sau grafică, L2 L1 L0 este o grupare de trei biţi ce reprezintă numărul liniei de matrici de 8×8 biţi (care în total sînt 24, T1 T0 este o grupare de doi biţi ce reprezintă numărul treimii în care se află linia de matrici referită, iar O4 O3 O2 O1 O0 este o grupare de cinci biţi ce reprezintă numărul matricii de 8×8 biţi (deci octetul ce reprezintă o linie de 8 biţi din matrice) dintr-un rînd (amintim că pe o linie sau pe un rînd există 32 coloane). Precizăm faptul important că ordinea de avansare a acestor grupe de biţi este următoarea: O, R, L, T; iar valorile limită ale acestor grupe de biţi sînt: O (00000—11111), R (000—111), L (000—111), T (00—10).

1.1.4.3 *Funcţiile de terminal*

Pe lîngă funcţia obişnuită de afişare a setului de caractere alfanumerice şi grafice, mai sînt implementate şi alte funcţii de terminal, care vor fi prezentate mai jos:

ESC,1 =1BH,31H — poziţionare absolută XY a cursorului
ESC,3 =1BH,33H — comutare „wraparound"/blocare verticală
ESC,4 =1BH,34H — comutare „wraparound"/blocare orizontală
ESC,5 =1BH,35H — auto line-feed inactiv/activ
ESC,A =1BH,41H — cursor în sus
ESC,B =1BH,42H — cursor în jos
ESC,C =1BH,43H — cursor la dreapta
ESC,D =1BH,44H — cursor la stînga
ESC,E =1BH,45H — ştergere ecran
ESC,H =1BH,48H — cursor în poziţia „home" (stînga sus)
ESC,I =1BH,49H — regres linie cu defilare în jos
ESC,J =1BH,4AH — ştergere pînă la sfîrşit de pagină
ESC,K =1BH,4BH — ştergere pînă la sfîrşitul liniei
ESC,L =1BH,4CH — înserare a unei linii vide
ESC,M =1BH,4DH — eliminarea liniei curente
ESC,N =1BH,4EH — activare afişare în invers video
ESC,O =1BH,4FH — revenire la afişare în video direct
ESC,R =1BH,52H — înserare spaţiu în poziţia curentă
ESC,S =1BH,53H — ştergere a caracterului curent din linie
ESC,Y =1BH,59H — poziţionare absolută YX a cursorului

CTRL/E =05H — eliminare a liniei curente
CTRL/F =06H — înserare a unei linii vide
CTRL/H =08H — cursor la stînga
CTRL/I =09H — tabulare orizontală
CTRL/J =0AH — LF avans linie
CTRL/M =0DH — CR retur car
CRTL/N =0EH — activare afişare în video invers
CTRL/O =0FH — revenire la afişare în video direct

CTRL/R =12H — înserare spaţiu în poziţia curentă
CTRL/U =15H — cursor la dreapta
CTRL/V =16H — ştergere pînă la sfîrşitul liniei
CTRL/W=17H — ştergere pînă la sfîrşitul paginii
CTRL/X =18H — ştergere ecran
CTRL/Y =19H — cursor în poziţia „home" (stînga sus)
CTRL/Z =1AH — cursor în sus
CTRL/9 =1BH — ESC iniţiază o secvenţă de escape
CTRL/B =02H — setează culoarea BORDER-ului (acumulatorul va conţine codul culorii dorite pentru BORDER)
CTRL/C =03h — setează octetul atribuit (ca mai sus)
CTRL/D =04h — programează interfaţa RS232
DEL =7FH — şterge caracterul precedent

1.1.4.4 *Interfaţa serială RS232*

Interfaţa serială RS232 este implementată soft şi deci nu este folosit nici un circuit specializat în acest scop. Aceasta e constituită din trei componente care asigură şi funcţiile pentru care a fost implementată: programarea parametrilor interfeţei, emisia şi recepţia. Utilizarea funcţiilor de emisie şi recepţie se face prin intrările **(BIOS) CONIN, CONOUT,** după modificarea în prealabil a octetului **IOBYTE** (corespunzător), sau numai în emisie prin intrarea **LIST.** Parametrii de funcţionare ai interfeţei se programează cu ajutorul utilitarului **RS232.COM** sau cu ajutorul funcţiei de terminal CTRL/D unde acumulatorul va conţine un cod ce are următoarea semnificaţie:

p	p	x	b	b	v	v	v

unde: viteza vvv =000 — 150 bauds
001 — 300 bauds
010 — 600 bauds
011 — 1200 bauds
100 — 2400 bauds
101 — 4800 bauds
110 — 9600 bauds
111 — 19200 bauds

nr. biţi bb= 00 — 5 biţi
01 — 6 biţi
10 — 7 biţi
11 — 8 biţi

XONN-XOFF x = 0 — nu
1 — da

paritate pp = 00 — fără paritate
01 — paritate indiferentă
10 — paritate pară
11 — paritate impară

1.2 EXECUŢIA PROGRAMELOR TRANZITORII

Programele tranzitorii sînt comenzi tranzitorii CP/M şi programe-utilizator de aplicaţii.

Orice program tranzitoriu se încarcă de pe disc în zona TPA şi se execută după cum va fi prezentat în continuare.

Utilizatorul comunică cu componenta CCP (deci cu sistemul CP/M) prin introducerea, după fiecare prompter CP/M (">") a unei linii de comandă. Fiecare linie de comandă are una din următoarele forme:

(1) comanda <CR>

(2) comanda specificator-fişier <CR>

(3) comanda specificator-fişier 1 specificator-fişier 2 <CR>

unde „comanda" este numele unei comenzi CP/M rezidente (ex: ERA, DIR, TYPE etc.) sau numele unei comenzi CP/M tranzitorii sau numele unui program-utilizator. Dacă „comanda" este numele unei comenzi CP/M, atunci această comandă este executată imediat. În caz contrar, CCP caută pe discul specificat (indicat înainte de comandă) sau pe discul instalat, un fişier cu numele:

comanda. COM

Dacă un astfel de fişier este găsit, atunci se presupune că el reprezintă imaginea-memorie a unui program care se execută în zona TPA şi care, implicit, se încarcă în memorie începînd de la adresa TBASE. Componenta CCP încarcă fişierul tip „.COM" de pe disc în memorie, începînd de la adresa TBASE şi îi predă controlul printr-o instrucţiune de tip „CALL". La sfîrşitul execuţiei programului, controlul poate reveni în CCP (printr-o instrucţiune de tip „RET") sau în CP/M (printr-o instrucţiune „JMP BOOT"). Dacă se doreşte ca la sfîrşitul execuţiei programului controlul să revină în CCP, atunci programul trebuie să nu suprascrie zona CBASE — FBASE. În caz contrar, programul poate să folosească memoria pînă la adresa FBASE — 1.

Dacă în linia de comandă există unul sau doi specificatori de fişier, atunci componenta CCP pregăteşte şi unul sau două „blocuri de control fişier" (FCB), în zona de memorie rezervată pentru „parametri sistem". Aceste FCB-uri sînt construite în formatul impus de FDOS pentru accesul la fişiere (vezi cap. 1.4).

Programele tranzitorii pot folosi:

— facilităţile CP/M de I/E pentru a comunica cu consola şi cu dispozitivele periferice, precum şi

— subsistemul de lucru cu discul, pentru accesul la fişiere rezidente pe acest suport.

Accesul din programe tranzitorii la sistemul de I/E al CP/M se face prin transmiterea către sistemul CP/M, prin punctul de intrare în FDOS (existent la adresa BOOT+0005H), a unui număr de rutină şi a unei adrese pentru informaţii specifice rutinei. După execuţia rutinei, FDOS întoarce o valoare ce indică modul de desfăşurare a operaţiei (operaţie desfăşurată corect sau codul de eroare (numeric), dacă aceasta a eşuat).

1.3 CONVENŢII PENTRU APELUL FUNCŢIILOR DE SISTEM CP/M

Sistemul CP/M pune la dispoziţia utilizatorilor o serie de rutine care pot fi apelate în cadrul programelor tranzitorii. Rutinele se împart în două categorii:

— rutine de I/E pentru periferice simple

— rutine de I/E pentru lucrul cu fişiere pe disc.

Rutinele de I/E pentru periferice simple sînt:

— citire caracter de la consolă
— scriere caracter la consolă
— citire/scriere directă la consolă
— citire caracter de la dispozitivul tip „READER"
— scriere caracter la dispozitivul tip „PUNCH"
— scriere caracter la dispozitivul tip „LIST"
— citire/modificare octet IOBYTE
— tipărire la consolă a unuişir de caractere sînt
— citire buffer consolă
— citire stare consolă.

Rutinele de I/E pentru lucrul cu fişiere pe disc sînt

— creare fişier
— deschidere fişier
— închidere fişier
— căutare în „director"
— modificare nume fişier
— ştergere fişier
— citire secvenţială sau directă a unui fişier
— scriere secvenţială sau directă a unui fişier
— modificare atribute fişier
— iniţializare „adresa DMA"
— iniţializare stare sistem discuri
— ş.a.

În ANEXA 1 este prezentată lista completă a rutinelor CP/M disponibile.

Accesul la rutinele FDOS se realizează prin transmiterea în:

— registrul „C" a numărului rutinei şi

perechea de registre „D&E" a unor parametri necesari rutinei.

Rutinele FDOS pot avea ca ieşiri valori pe un octet (în registrul „A") sau pe doi octeţi (în perechea de registru „H&L").

OBSERVAŢII: a. Pentru rutinele care au ca ieşiri valori pe doi octeţi, aceste valori se găsesc şi în registrele „A" şi „B" (i.e. (A)=L şi (B)=H)

b. Convenţiile de apel al rutinelor CP/M respectă standardele PL/M de comunicare parametri.

c. Există rutine CP/M care folosesc doar registrul „E" pentru transmiterea unor parametri necesari apelului lor

d. Există rutine CP/M care nu necesită parametri (apelul lor presupune doar transmiterea, prin registrul „C", a numărului rutinei)

e. Există rutine CP/M care nu au ieşiri.

Rezultă că secvenţa standard necesară pentru apelul unor rutine CP/M este:

```
BDOS    EQU     0005H


        MVI     C, număr rutină
[       LXI     D, parametrii specifici rutinei ]
[       MVI     E; parametru specific rutinei]
        CALL    BDOS        ;apel rutină prin punctul de intrare
                            ; în FDOS
```

OBSERVAŢIE: Liniile cuprinse între [,] reprezintă linii opţionale, dependente de tipul rutinei.

S-a arătat în capitolul 1.2 că după încărcarea de pe disc, în memorie, a unui program tranzitoriu, componenta CCP îi predă acestuia controlul printr-o instrucţiune de tip „CALL". Execuţia programului tranzitoriu începe avînd SP-ul poziţionat pe o stivă cu o capacitate de 8 nivele (16 octeţi), în care există înscrisă doar adresa de revenire în CCP (7 nivele sînt încă libere). Deşi această stivă nu este de obicei folosită de către programele tranzitorii (majoritatea acestora rezervîndu-şi o stivă proprie şi revenind în CCP printr-o instrucţiune de tip „JMP BOOT"), totuşi este util de cunoscut faptul că ea este suficient de mare pentru a realiza apeluri de rutine CP/M. Acest lucru este posibil întrucît componenta FDOS la fiecare intrare într-o rutină de sistem, comută SP-ul pe o stivă locală, neafectînd astfel stiva iniţială a programului. Programul în limbaj de asamblare de mai jos reprezintă un exemplu în acest sens, el realizînd citirea unor caractere de la consolă, pînă la întîlnirea unui caracter „*" care determină întoarcerea controlului în CCP:

```
BDOS    EQU     0005H    ; punct de intrare standard în rutinele
                         ; CP/M
CONIN   EQU     1        ; rutina „Console Input"
        ORG     100H     ; adresa de bază pentru TPA
NEXTC
        MVI     C,CONIN  ; pregătire apel rutina CONIN
        CALL    BDOS     ; citire caracter de la consolă cu pre-
                         ; luarea caracterului în registrul „A"
        CPI     '*'      ; test pentru sfîrşit de prelucrare
        JNZ     NEXTC    ; reluare prelucrare dacă nu e '*'
        RET              ; revenire în CCP
        END
```

1.4. PARTICULARITĂŢI ÎN UTILIZAREA RUTINELOR CP/M DE LUCRU CU FIŞIERE PE DISC

Pentru lucrul cu discul flexibil sistemul CP/M implementează, pe fiecare volum disc, o structură de fişiere identificate prin nume. Fiecare unitate de disc este, din punct de vedere logic, distinctă, avînd o zonă rezervată pentru **„director"** şi o altă zonă pentru **fişierele de date**. Fiecare fişier are asociat un identificator alcătuit din:

- codul pentru selectarea unităţii de disc (o literă A—P)
- numele (alcătuit din 1—8 caractere ASCII diferite de spaţiu)
- extensia (tipul) fişierului (alcătuită din 0—3 caractere ASCII diferite de spaţiu)

Extensiile definesc categoria generică din care face parte un anumit fişier, în timp ce numele identifică în mod unic fişierul în cadrul categoriei respective. Astfel, sistemul CP/M utilizează următoarele **extensii standard:**

- **ASM** pentru fişiere sursa în limbaj de asamblare tratabile cu asamblorul ASM sau MAC
- **PRN** pentru fişiere listing
- **HEX** pentru fişiere hexa
- **BAS** pentru fişiere sursă în limbaj BASIC
- **INT** pentru fişiere cod-obiect intermediar
- **COM** pentru fişiere cod-obiect direct executabil
- **REL** pentru fişiere cod-obiect relocabil
- **COB** pentru fişiere sursa în limbaj COBOL
- **FOR** pentru fişiere sursa în limbaj FORTRAN
- **MAC** pentru fişiere sursa în limbaj de ansamblare tratabile cu asamblorul M80
- **BAK** pentru fişiere ce reprezintă versiuni anterioare într-un proces de editare texte
- **$$$** pentru fişiere temporare
- **ş.a.**

Fişierele sursă sînt tratate ca o secvenţă de caractere ASCII, în care fiecare „linie" din fişier se termină prin secvenţa de caractere <CR> <LF> (0DH 0AH). Astfel, o înregistrare CP/M (de 128 de octeţi) poate conţine mai multe linii de text sursă. Sfîrşitul unui fişier ASCII este indicat prin caracterul CTRL/Z (1AH) sau prin „sfîrşitul fizic" de fişier, detectat de către rutina CP/M de citire. Caracterele CTRL/Z existente într-un fişier cod-obiect (de exemplu, în fişiere tip COM) sînt ignorate, sfîrşitul de fişier fiind detectat de rutina CP/M de citire.

Orice fişier CP/M este o secvenţă de maximum 65536 înregistrări, de cîte 128 octeţi fiecare, numerotate de la 0 la 65535. Deşi din punct de vedere logic înregistrările într-un fişier sînt contigue totuşi, fizic (pe disc) ele pot să nu fie contigue.

Fiecare fişier este, intern, împărţit în segmente de cîte 16 KB, denumite **„extensii logice"**. În cadrul fiecărei „extensii logice" există 128 de înregistrări (128*128 B=16 KB) numerotate de la 0 la 127 (00H—7FH). Se observă că în cadrul unei „extensii logice" contorul de înregistrări poate fi reprezentat pe 8 biţi. Informaţiile privind fiecare „extensie logică" a unui fişier ocupă spaţiu în „directorul" discului respectiv. O „extensie logică" (16

KB) este formată din mai multe blocuri de alocare. Un bloc de alocare reprezintă spaţiul disc minim ce poate fi alocat unui fişier. Un bloc de alocare are minimum 1 KB şi maximum 16 KB; dimensiunea blocului de alocare este stabilită la generarea sistemului CP/M.

Pentru utilizarea rutinelor CP/M de lucru cu fişiere pe disc trebuie respectate următoarele convenţii:

— informaţiile de identificare a oricărui fişier se transmit către rutinele FDOS într-un format standard, şi anume sub forma unui **„bloc de control fişier" (File Control Block=FCB)**. Dimensiunea FCB depinde de tipul accesului la fişier (este de 33 de octeţi pentru acces secvenţial şi de 36 de octeţi pentru acces direct). Adresa FCB-ului se transmite în general prin registrele D&E.

— orice operaţie de citire/scriere date într-un fişier se realizează asupra unei înregistrări de **128 de octeţi.**

— adresa de început a zonei de memorie (de 128 de octeţi) utilizată ca buffer în operaţiile de citire/scriere se numeşte **„adresa DMA"**. Această adresă nu se transmite ca parametru, ea fiind iniţializată de către sistemul CP/M sau de către o rutină CP/M specială (rutina 26).

— se numeşte **„disc selectat"** acea unitate de disc care a fost activată prin:

— acţiunea componentei CCP (**discul instalat** prin CCP este în momentul lansării unui program în TPA **„disc selectat"**) sau

— prin rutina CP/M de selectare disc (rutina 14)

— se numeşte **„disc activ"** acea unitate de disc, care de la ultima iniţializare/reiniţializare a sistemului CP/M sau de la ultima operaţie de iniţializare stare sistem discuri (rutina 13), a făcut obiectul unei selecţii:

— **explicite** (prin CCP sau prin rutina CP/M de selectare disc (rutina 14)) sau

— **implicite** (printr-o rutină de deschidere sau creare a unui fişier **cu octet 00 din FCB diferit de zero).**

OBSERVAŢIE: Sistemul CP/M foloseşte intern pentru **FCB** zona de memorie **005CH — 007FH** (36 octeţi), iar ca **„adresă DMA"** adresa **0080H** (bufferul pentru operaţii de citire/scriere este de la adresa 0080H pînă la 00FFH (128 octeţi)). Utilizatorul poate să folosescă în programe aceste zone de memorie pentru FCB-ul, respectiv pentru bufferul propriu.

Structura standard a unui FCB este următoarea:

— **octetul 00** — **codul unităţii de disc** pe care se găseşte fişierul, respectiv:

00H — pentru discul selectat

01H — pentru unitatea „A"

.

.

.

10H — pentru unitatea „P"

— octeţii 01 —08 — numele fişierului exprimat în ASCII (caractere majuscule, cu B7=0); dacă numele fişierului are mai puţin de 8 caractere, atunci el **trebuie completat la dreapta cu blancuri.**

— octeţii 09—11 — **extensia** (tipul), fişierului exprimat în ASCII (caractere majuscule); dacă extensia are mai puţin de 3 caractere, atunci acest cîmp **trebuie completat la dreapta cu blancuri.**
Dacă fişierul este protejat la scriere (R/O) atunci B7 din octetul 09 este egal cu 1; altfel, acest bit este egal cu 0.
Dacă fişierul este invizibil (SYS) atunci B7 din octetul 10 este 1; altfel, acest bit este egal cu 0.

— octetul 12 — **numărul curent al „extensiei logice"** a fişierului; de obicei acest octet este setat de utilizator pe 00H.

— octetul 13 — rezervat pentru sistem

— octetul 14 — rezervat pentru sistem; acest octet este setat pe 00H atunci cînd se execută operaţii de tip OPEN, MAKE, SEARCH.

— octetul 15 — **contor de înregistrări în cadrul „extensiei logice" curente** (ia valori între 00H şi 7FH); acest cîmp este completat de către sistem.

— octeţii 16—31 — rezervaţi pentru sistem (ei vor fi completaţi de către sistem)

— octetul 32 — **numărul înregistrării din „extensia logică" curentă**; se foloseşte în accesul secvenţial la fişiere; în mod normal acest octet este setat de către utilizator pe 00H la deschiderea fişierului.

— octeţii 33—35 — reprezintă un parametru opţional folosit numai în accesul direct la fişiere. El indică **numărul înregistrării de scris/citit** (are valori între 0000H şi FFFFH cu posibilitate de depăşire în octetul 35). Octeţii 33 şi 34 reprezintă o valoare pe 16 biţi cu partea cea mai puţin semnificativă în octetul 33 şi cea mai semnificativă în octetul 34.

Fiecare fişier care trebuie exploatat prin CP/M trebuie să aibă un FCB propriu, care să furnizeze, pentru fiecare operaţie cu fişierul informaţii privind numele şi alocarea acestuia. Orice acces la un fişier implică iniţializarea de către utilizator (programator) a FCB-ului corespunzător, respectiv prin înscrierea în octeţii 00—11 a specificatorului fişierului şi prin umplerea cu 00H a restului de octeţi (12—31/35). Informaţiile din FCB-urile corespunzătoare fişierelor de pe un disc se găsesc înregistrate în „directorul" discului respectiv şi sînt aduse în memoria internă înainte ca utilizatorul să înceapă lucrul asupra fişierului/fişierelor (vezi rutinele OPEN, MAKE). Copia din

memorie a FCB-ului este actualizată pe măsură ce au loc operații asupra fişierului, iar la terminarea lucrului cu acesta ea este înregistrată pe disc (vezi rutina CLOSE).

Atunci cînd o comandă (program) se lansează în execuție prin:

comanda specificator-fişier 1<CR>

comanda specificator-fişier 1 specificator-fişier 2 <CR>

componenta CCP construieşte (după cum s-a arătat în cap. 1.2) primii 16 octeţi din două FCB-uri pornind de la specificatorul/specificatorii de fişier prezenți în linia de comandă (după numele comenzii). Automat, CCP completează (dacă este cazul) numele şi extensia fişierelor cu blancuri. Primul FCB este construit la adresa 005CH şi poate fi folosit ca atare pentru operații ulterioare asupra fişierului „specificator-fişier". Al doilea FCB este construit în octeții 16—31 din primul FCB (adică de la adresa 006CH) şi **trebuie să fie mutat într-o altă zonă de memorie înainte de utilizarea lui.** Dacă de exemplu, utilizatorul introduce comanda:

PROGNAME B:X.ZOT Y.ZAP <CR>

fişierul PROGNAME.COM de pe discul instalat va fi încărcat în zona TPA, iar blocul de control de la adresa 005CH va fi inițializat astfel:

octetul 00 =02H (cod unitate „B")
octeții 01—08='X'
octeții 09—11='ZOT'
octeții 12—15=00H
octetul 16 =00H (cod disc selectat, care în acest caz este chiar discul instalat)
octeții 17—24='Y'
octeții 25—27='ZAP'
octeții 28—31=00H

OBSERVAȚIE: Programatorul trebuie să salveze conținutul celui de-al doilea FCB (cei 16 octeţi începînd de la adresa 006CH) înainte de a deschide fişierul corespunzător primului FCB (de la adresa 005CH), întrucît prin deschiderea acestuia informațiile referitoare la cel de-al doilea fişier vor fi şterse (suprascrise) de către sistem (de către rutina OPEN).
Dacă într-o linie de comandă CP/M nu apare nici un specificator-fişier atunci zonele 005DH—0067H şi 006DH—0077H vor conține blancuri.
Componenta CCP asigură automat transformarea minusculelor în majuscule.

O caracteristică importantă pentru utilizator **a sistemului CP/M** este şi aceea că după recepționarea unei linii de comandă **el păstrează la adresa 0080H un buffer pentru consolă**, în care există o copie a conținutului liniei de comandă exceptînd numele comenzii. Astfel, pentru exemplul considerat anterior, bufferul de la adresa 0080H va avea următorul conținut:

octetul 00 =0EH (numărul de caractere utile din linia de comandă exceptînd numele comenzii)
octetul 01 =' '
octeții 02—08='B:X.ZOT'

octetul 09 = ' '
octeţii 10–14='Y.ZAP'

Este, şi în acest caz, sarcina utilizatorului de a extrage informaţiile din acest buffer, înainte de a executa orice operaţie asupra unui fişier, operaţie prin care această zonă este suprascrisă (iniţial „adresa DMA" este egală cu 0080H, adică tocmai adresa de început a acestui buffer consolă).

1.5 PREZENTAREA RUTINELOR CP/M

RUTINA 0: Reiniţializare sistem CP/M
(System Reset)

Intrări:

registrul C: 00H

Efect: întoarce controlul din programul utilizator în CP/M; această funcţie are acelaşi efect ca „JMP 0000H".

RUTINA 1: Citire caracter de la consolă
(Console Input)

Intrări:

registrul C: 01H

Ieşiri:

registrul A: codul unui caracter ASCII

Efect: preia un caracter de la consolă şi-l transmite în registrul „A". **Toate caracterele tipăribile şi în plus <CR>, < LF> şi <BS> (CTRL/H) sînt transmise în ecou la consolă.** De asemenea caracterul CTRL/I (TAB) mută cursorul în următoarea poziţie de tabulare. Restul de caractere netipăribile nu sînt transmise în ecou la consolă. Rutina aşteaptă un timp nelimitat pînă cînd se tastează un caracter la consolă.

RUTINA 2: Scriere caracter la consolă
(Console Output)

Intrări:

registrul C: 02H
registrul E: codul caracterului ASCII

Efect: transmite la consolă caracterul specificat prin registrul „E". Caracterele „TAB" (CRTL/I) sînt expandate iar caracterul CTRL/S este

interpretat drept stop defilare. Reluarea defilării, după CTRL/S se face cu orice caracter diferit de CTRL/C, care reiniţializează sistemul CP/M.

RUTINA 3: Citire caracter de la dispozitivul „Reader" curent (Reader Input)

Intrări:

registrul C: 03H

Ieşiri:

registrul A: codul unui caracter ASCII

Efect: preia un caracter de la dispozitivul RDR: curent şi-l depune în registrul „A". Rutina aşteaptă un timp nelimitat preluarea caracterului de la RDR:.

RUTINA 4: Scriere caracter la dispozitivul „Punch" curent (Punch Output)

Intrări

registrul C: 04H
registrul E: codul unui caracter ASCII

Efect: transmite la dispozitivul PUN: curent caracterul specificat prin registrul „E".

RUTINA 5: Scriere caracter la dispozitivul „List" curent (List Output)

Intrări:

registrul C: 05H
registrul E: codul unui caracter ASCII

Efect: transmite la dispozitivul LST: curent caracterul specificat prin registrul „E".

RUTINA 6: Citire/Scriere directă la consolă (Direct Console I/O)

Intrări:

registrul C: 06H
registrul E: — 0FFH (pentru citire de la consolă)
— codul unui caracter ASCII (pentru scriere la consolă)

Ieşiri: registrul A: codul unui caracter ASCII sau octet de stare

Efect: dacă registrul „E" este egal cu 0FFH, atunci rutina realizează citirea (fără ecou) a unui caracter de la consolă. Registrul „A" va conţine codul caracterului ASCII introdus sau 00H, dacă nu s-a introdus nici un caracter.

OBSERVAŢII: Rutina nu aşteaptă nelimitat introducerea unui caracter de la consolă (ea întoarce imediat (A)=00H, dacă în registrul de interfaţă al consolei nu există nici un caracter disponibil). Este indicat ca utilizatorul să aştepte prin program introducerea unui caracter de la consolă.
Nu sînt active în acest caz, caracterele de editare ale sistemului CP/M (ex: DEL, TAB, CTRL/R etc.).

Dacă registrul „E" conţine codul unui caracter ASCII, atunci rutina realizează scrierea la consolă a caracterului respectiv.
Rutina 6 nu trebuie să fie folosită împreună cu alte rutine CP/M care realizează intrări/ieşiri cu consola (rutinele 1, 2, 9, 10 şi 11).

RUTINA 7: Citire octet IOBYTE
(Get I/O Byte)
Intrări:
registrul C: 07H
Ieşiri:
registrul A: valoarea curentă a octetului IOBYTE

Efect: citeşte octetul de la adresa 0003H şi îl plasează în registrul „A".

RUTINA 8: Modificare octet IOBYTE
(Set I/O Byte)
Intrări:
registrul C: 08H
registrul E: valoare pentru octetul IOBYTE

Efect: scrie conţinutul registrului „E" la adresa 0003H, modificînd astfel configuraţia de I/E curentă.

RUTINA 9: Tipărire la consolă a unui şir de caractere
(Print String)
Intrări:
registrul C: 09H
registrele D&E: adresa şirului de caractere

Efect: tipăreşte la consolă şirul de caractere ASCII a cărui adresă de început este specificată în registrele „D&E". Tipărirea se termină atunci

cînd s-a întîlnit caracterul „$". Rutina tratează caracterele TAB (CTRL/I) întîlnite, mutînd cursorul în următoarea poziţie de tabulare. La fel ca în rutina 2, se face verificare pentru caracterul CTRL/S (stop defilare).

RUTINA 10: Citire buffer consolă
(Read Console Buffer)

Intrări:

registrul C: 0AH
registrele D&E: adresa buffer

Efect: rutina permite citirea unei linii introduse de la consolă cu transferarea conţinutului ei într-o zonă de memorie a cărei adresă de început este dată în registrele „D&E".

O linie editată la consolă se consideră terminată atunci cînd s-a introdus caracterul ⟨CR⟩ sau caracterul ⟨LF⟩ sau atunci cînd s-a depăşit capacitatea bufferului consolei **specificată de utilizator în primul octet din buffer.** Rutina aduce în bufferul a cărui adresă este dată în registrele „D&E" următorul conţinut:

— **octetul 00** = **numărul maxim de caractere din bufferul consolei** (cu valori între 1 şi 255); acest cîmp este **iniţializat de către utilizator înaintea apelului rutinei 10**

— **octetul 01** = numărul de caractere introduse în linie (fără <CR> şi <LF>)

— **octeţii 02-n** = caracterele din linia de editare (c1, c2, c3, ..., cn)

OBSERVAŢII: Dacă numărul de caractere din linia de editare este mai mic decît numărul maxim de caractere din buffer, atunci după ultimul caracter citit din linia de editare (i.e. caracterul „cr") şi pînă la poziţia corespunzătoare ultimului caracter posibil în buffer, vor exista (în buffer) o serie de caractere fără semnificaţie pentru utilizator (ele reprezintă un rest neiniţializat din bufferul consolei).

În timpul introducerii de la consolă a liniei sînt active, pentru corecţii, caracterele de editare ale sistemului CP/M:

RUBOUT/DEL — şterge din bufferul de intrare şi redă în ecou ultimul caracter introdus de la consolă

CTRL/C — reîncărcarea sistemului de operare

CTRL/E — indică sfîrşitul fizic al unei linii; cursorul se poziţionează pe începutul liniei dar linia nu se transmite decît atunci cînd se tastează ⟨CR⟩

CTRL/H — introduce în bufferul de intrare un caracter „back space" care are ca efect întoarcerea cursorului pe ecran cu o poziţie

CTRL/J — este echivalent unui caracter<LF>și reprezintă sfîrșitul unei linii

CTRL/M — este echivalent unui caracter <CR> și reprezintă sfîrșitul unei linii

CTRL/R — tipărește la consolă pe linia imediat următoare conținutul curent al bufferului de intrare. Prin acest caracter se poate vizualiza conținutul curent al unei linii în care s-au efectuat corecții prin RUBOUT (DEL)

CTRL/U — șterge integral linia introdusă de la consolă

CTRL/X — identic cu CTRL/U

RUTINA 11: Citire stare consolă (Get Console Status)
Intrări:
registrul C: 0BH
Ieșiri:
registrul A: stare consolă

Efect: rutina verifică dacă s-a introdus un caracter de la consolă sau nu. Dacă în registrul de interfață al consolei există un caracter disponibil, atunci rutina întoarce în registrul „A" valoarea 0FFH. În caz contrar, în registrul „A" se va afla valoarea 00H.

RUTINA 12: Citire versiune sistem (Return Version Number)
Intrări:
registrul C: 0CH
Ieșiri:
registrele H & L: număr de versiune

Efect: rutina întoarce în registrele „H & L" o valoare egală cu numărul de versiune al sistemului CP/M sub care se lucrează, respectiv (H)=00H iar (L)=numărul de versiune (ex: 22H pentru versiunea 2.2).

RUTINA 13: Inițializare stare sistem discuri (Reset Disk System)
Intrări:
registrul C: 0DH

Efect: rutina dezactivează logic toate unitățile de disc (le acordă atributul R/W), asignează ca disc selectat unitatea „A" și stabilește ca „adresa

DMA“ adresa 0080H. Rutina poate fi folosită atunci cînd o aplicaţie necesită schimbări de volume disc fără a se reiniţializa sistemul CP/M (prin CTRL/C)

RUTINA 14: Selectare disc
(Select Disk)

Intrări:

registrul C: 0EH
registrul E: număr unitate selectată

Efect: rutina desemnează unitatea specificată în registrul „E“ ca „disc selectat“. Numărul unităţii de disc se specifică prin valorile: 00H pentru unitatea „A“, 01H pentru unitatea „B“,..., 0FH pentru unitatea „P“. În urma execuţiei rutinei, unitatea specificată în registrul „E“ este trecută în starea „activ“ (disc activ) care încarcă „directorul“ volumului respectiv; unitatea rămîne în această stare pînă la o nouă iniţializare sau reiniţializare a sistemului CP/M sau pînă la o nouă operaţie de „iniţializare stare sistem discuri“ (rutina 13). **Dacă în timp ce o unitate este „activă“ se fac schimbări de volume disc, atunci automat unitatea este desemnată de către sistem ca R/O (vezi şi rutina 28).**

OBSERVAŢIE: Toate FCB-urile care au primul octet egal cu 00H se referă implicit la fişiere care se găsesc pe discul selectat.

RUTINA 15: Deschidere fişier (Open File)

Intrări:

registrul C: 0FH
registrele D & E: adresa FCB

Ieşiri

registrul A: octet de stare

Efect: rutina realizează activarea unui fişier care se găseşte în „directorul“ discului specificat prin octetul 00 din FCB şi care aparţine utilizatorului curent. Adresa FCB-ului fişierului de deschis este dată prin registrele „D & E“.

Programul FDOS caută în directorul discului specificat o intrare identică cu valoarea octeţilor 1—12 din FCB.

OBSERVAŢIE: În FCB octeţii 12 şi 32 trebuie setaţi de către utilizator pe 00H, înaintea apelului rutinei 15.

Dacă programul FDOS găseşte o astfel de intrare, atunci informaţiile din „director“ corespunzătoare ei sînt copiate în octeţii 15—31 din FCB, permiţîndu-se astfel accesul la fişier pentru operaţii ulterioare de citire/scriere.

Rutina intoarce în registrul „A“ o valoare 0—3, dacă operaţia de deschidere s-a efectuat corect şi o valoare egală cu 255 (0FFH), dacă aceasta a eşuat.

OBSERVAŢIE: Programatorul nu trebuie să efectueze operaţii asupra unui fişier. decît după ce s-a realizat corect deschiderea sa.

Există posibilitatea ca în cadrul FCB-ului, în octeţii 1—11 să apară un specificator-multiplu de fişier adică să apară caractere „?" (care înlocuiesc orice caracter în poziţia respectivă). În acest caz, programul FDOS caută în „director" prima intrare care corespunde specificatorului-multiplu de fişier din FCB.

OBSERVAŢIE: Dacă fişierul deschis prin această rutină urmează să fie exploatat secvenţial, începînd cu primul său articol, atunci utilizatorul trebuie să seteze octetul 32 din FCB pe 00H (pentru ca prima citire/scriere să se aplice asupra primei înregistrări din fişier).

RUTINA 16: Închidere fişier (Close File)

Intrări:

registrul C: 10H
registrele D & E: adresa FCB

Ieşiri:

registrul A: octet de stare

Efect: rutina realizează reversul rutinelor 15 (OPEN) şi 22 (MAKE). Astfel, presupunînd că FCB-ul a cărui adresă este specificată în registrele „D & E" a fost activat anterior printr-o rutină de „deschidere fişier" (rutina 15) sau de „creare fişier" (rutina 22), rutina de „închidere fişier" înregistrează FCB-ul curent în „directorul" discului specificat, actualizînd astfel intrarea din „director" corespunzătoare fişierului respectiv.

Rutina întoarce în registrul „A" o valoare egală cu 0—3 dacă operaţia de închidere s-a desfăşurat corect sau o valoare egală cu 255 (0FFH) dacă numele fişierului din FCB nu a fost găsit în „director".

Închiderea fişierelor care au fost exploatate doar în citire este opţională. Numai fişierele în care s-au efectuat operaţiile de scriere trebuie închise (pentru a actualiza în „director" informaţiile referitoare la acele fişiere).

Dacă în FCB-ul fişierului de închis apare un specificator multiplu (i.e. caractere „?"), atunci rutina va efectua căutarea în „director" aşa cum face rutina 15.

RUTINA 17: Caută în „director" prima intrare (Search for First)

Intrări:

registrul C: 11H
registrele D & E: adresa FCB

Ieşiri:

registrul A: octet de stare

Efect: rutina caută în „director" prima intrare care corespunde valorilor octeţilor 0—12 din FCB-ul a cărui adresă este dată în registrele „D & E". Rutina întoarce în registrul „A" valoarea 255 (0FFH) dacă nu a găsit o astfel de intrare sau o valoare cuprinsă între 0—3 dacă a găsit-o. Dacă în „director"

a fost găsită o intrare identică cu specificatorul-fişierului din FCB, atunci zona de memorie a cărei adresă este „adresa DMA" va fi completată cu o înregistrare de „director" (128 octeţi) şi-anume cu acea înregistrare din „director" care conţine intrarea respectivă. Adresa relativă a intrării, în cadrul înregistrării de „director", este egală cu (A) 32 (i.e. conţinut de registru „A" rotit spre stînga cu 5 biţi sau „ADD A" de 5 ori). Programele de aplicaţii pot extrage, pe baza acestei adrese relative, din bufferul care conţine înregistrarea de „director", informaţiile din „director" relative la intrarea găsită.

Dacă FCB-ul conţine un specificator-multiplu (i.e. apar caractere „?" în poziţiile 1—12), atunci rutina întoarce PRIMA intrare din „director" care satisface specificatorul. Dacă octetul 00 din FCB conţine caracterul „?", atunci rutina întoarce **automat prima intrare din „directorul" discului selectat** (indiferent de numărul utilizatorului căruia îi aparţine intrarea respectivă, indiferent de continutul acestei intrări şi indiferent dacă intrarea este ştearsă sau nu).

RUTINA 18: Caută în „director" următoarea intrare (Search for Next)

Intrări:

registrul C: 12H

Ieşiri:

registrul A: octet de stare

Efect: această rutină este similară rutinei 17 cu excepţia faptului că „directorul" discului specificat nu se investighează de la începutul sau (ca în toate celelalte rutine), ci se caută intrarea corespunzătoare FCB-ului începînd de la ultima intrare din „director" găsită.

Rutina întoarce în registrul „A" (ca şi rutina 17), valoarea 255 (0FFH), dacă nu se mai găseşte în „director" nici o intrare identică cu FCB-ul specificat.

RUTINA 19: Ştergere fişier (Delete File)

Intrări:

registrul C: 13H
registrele D&E: adresa FCB

Ieşiri:

registrul A: octet de stare

Efect: rutina realizează ştergerea unuia sau mai multor fişiere, specificate prin FCB-ul a cărui adresă este dată în registrele „D&E". FCB-ul poate conţine un specificator-individual de fişier sau un specificator-multiplu de fişier (pot apare caractere „?" în zona de nume sau de extensie a fişierului, dar nu şi în zona pentru numele unităţii de disc pe care se găseşte fişierul aşa cum se putea în rutinele 17 şi 18).

Rutina întoarce în registrul „A" valoarea 255 (0FFH), dacă fişierul/fişierele specificate în FCB nu au fost găsite, şi o valoare 0—3, dacă operaţia de ştergere s-a efectuat normal.

RUTINA 20: Citire secvenţială
(Read Sequential)

Intrări:

registrul C: 14H
registrele D & E: adresa FCB

Ieşiri:

registrul A: octet de stare

Efect: presupunînd că FCB-ul a cărui adresă este specificată în registrele „D & E" a fost activat printr-o rutină de „deschidere fişier" (rutina 15) sau de „creare fişier" (rutina 22), rutina „citire secvenţială" realizează citirea din fişier a **următoarei înregistrări de 128 de octeţi** şi transferarea ei în memorie, într-o zonă a cărei adresă este „adresa DMA". **Numărul înregistrării** din cadrul „extensiei logice" curente **este specificat prin octetul 32** din FCB. După citire, valoarea acestui octet va fi automat incrementată cu 1. Dacă valoarea rezultată în octetul 32 depăşeşte 127 (7FH) atunci, automat, următoarea „extensie logică" a fişierului este deschisă şi octetul 32 ia valoarea 00H, fiind astfel pregătit pentru următoarea operaţie de citire. Dacă operaţia de citire s-a efectuat normal, atunci registrul „A" va avea valoarea 00H; în caz contrar, adică atunci cînd nu mai există date în fişier (s-a atins sfîrşitul fişierului!), registrul „A" va avea o valoare diferită de 00H.

RUTINA 21: Scriere secvenţială
(Write Sequential)

Intrări:

registrul C: 15H
registrele D & E adresa FCB

Ieşiri:

registrul A: octet de stare

Efect: presupunînd că FCB-ul a cărui adresă este specificată în registrele „D & E" a fost activat printr-o operaţie de „deschidere fişier" (rutina 15) sau „creare fişier" (rutina 22) anterioară, rutina „scriere secvenţială" realizează scrierea în fişier a unei înregistrări de 128 de octeţi. Înregistrarea de scris este luată din memorie, de la o adresă egală cu „adresa DMA" şi este plasată în fişier în poziţia dată de valoarea octetului 32 din FCB (numărul înregistrării în cadrul „extensiei logice" curente). După scrierea înregistrării în fişier, conţinutul octetului 32 din FCB este automat incrementat cu 1. Dacă în urma incrementării rezultă o depăşire (o valoare mai mare de 127 (i.e. 7FH)) atunci, automat, este deschisă următoarea „extensie logică" a fişierului şi octetul 32 din FCB este iniţializat cu 00H, în vederea unor operaţii de scriere ulterioare. Operaţia de „scriere secvenţială" poate avea loc şi în cadrul unor fişiere deja create corect, caz în care înregistrările ce se scriu se vor suprapune peste cele existente, practic înlocuindu-le pe cele vechi.

Rutina întoarce în registrul „A" valoarea 00H dacă operaţia de scriere a decurs normal sau o valoare diferită de 00H, dacă operaţia de scriere a eşuat datorită lipsei de spaţiu pe disc.

RUTINA 22: Creare fişier
(Make File)

Intrări:

registrul C: 16H
registrele D & E: adresa FCB

Ieşiri:

registrul A: octet de stare

Efect: rutina are acelaşi efect ca şi rutina „deschidere fişier" (rutina 15), cu excepţia faptului că în acet caz, FCB-ul trebuie să conţină numele unui fişier care nu există în „directorul" discului specificat.

Programul FDOS creează intrarea din „director" corespunzătoare FCB--ului şi iniţializează atît FCB-ul cît şi „directorul" discului, forţînd lungimea fişierului pe 0.

OBSERVAŢIE: Programul trebuie să evite duplicarea numelor fişierelor în „director", respectiv trebuie să se asigure că în „director" nu există un alt fişier cu nume identic cu cel al fişierului de creat. În acest scop, este indicat ca el să efectueze anterior rutinei 22 o operaţie de „ştergere fişier" (rutina 19).

Rutina 22 întoarce în registrul „A" o valoare 0—3 dacă operaţia s-a desfăşurat normal sau o valoare 255 (0FFH) dacă nu mai există spaţiu în „directorul" discului. Rutina 22 are ca efect secundar şi activarea FCB-ului, astfel încît nu mai este necesară o operaţie ulterioară de „deschidere fişier".

RUTINA 23: Schimbare nume fişier
(Rename File)

Intrări:

registrul C: 17H
registrele D & E: adresa FCB

Ieşiri:

registrul A: octet de stare

Efect: rutina realizează schimbarea numelui unui fişier. Rutina utilizează FCB-ul adresat prin registrele „D & E" astfel:

— primii 16 octeţi din FCB reprezintă numele vechi al fişierului
— ultimii 16 octeţi din FCB reprezintă numele nou al fişierului
— octetul 00 din FCB reprezintă numele unităţii pe care se găseşte fişierul de redenumit (octetul 16 din FCB trebuie să fie 00H).

Rutina întoarce în registrul „A" o valoare 0—3 dacă operaţia s-a desfăşurat normal sau valoarea 255 (0FFH) dacă nu s-a găsit în „directorul" discului specificat un fişier cu nume identic cu cel al fişierului de redenumit.

RUTINA 24: Citire vector de unități-disc active
(Return Log-in Vector)

Intrări:

registrul C: 18H

Ieșiri:

registrele H & L vectorul de unități-disc active

Efect: rutina analizează care din unitățile de disc A—P este „activă", respectiv care din aceste unități a fost activată:

explicit printr-o rutină de „selectare disc" (rutina 14)
implicit printr-o operație de deschidere/creare fișier (cu valoare diferită de 00H in octetul 00 din FCB).

Pentru unitățile de disc active, rutina întoarce o valoare logică „1", iar pentru cele care nu sînt active o valoare logică „0". Bitul B0 din registrul „L" reprezintă starea unității „A", iar bitul B7 din registrul „H" reprezintă starea unității „P". Astfel, prin registrele „H&L" (respectiv B &A) rutina întoarce un vector ce indică starea tuturor unităților A—P.

RUTINA 25: Citire număr disc selectat
(Return Current Disk)

Intrări:

registrul C: 19H

Ieșiri:

registrul A: numărul discului selectat

Efect: rutina întoarce în registrul „A" numărul „discului selectat". Acest număr este 00H pentru unitatea „A" și ... 0FH pentru unitatea „P".

RUTINA 26: Modificare „adresa DMA"
(Set DMA Address)

Intrări:

registrul C: 1AH
registrele D & E: adresa DMA

Efect: rutina permite modificarea „adresei DMA", adică a adresei bufferului de 128 octeți folosiți în operațiile de citire/scriere fișiere. În general, „adresa DMA" stabilită la inițializarea CP/M, la reinițializarea CP/M precum și după o operație de „inițializare stare sistem discuri" (rutina 13), este adresa 0080H. Rutina permite comutarea acestei adrese pe orice altă adresă (dată în registrele „D & E"), permițînd astfel localizarea bufferului de 128 de octeți in orice zonă de memorie.

Rutina stabilește „adresa DMA" ca fiind egală cu adresa specificată în registrele „D & E". Noua valoare pentru „adresa DMA" este valabilă pînă la:

— o inițializare sau reinițializare a sistemului CP/M
— un alt apel al rutinei 26
— o operație de „inițializare stare sistem discuri" (rutina 13).

RUTINA 27: Citire adresa vector de alocare
(Get ADDR (Alloc))

Intrări:

registrul C: 1BH

Ieșiri:

registrele H & L: adresa vectorului de alocare

Efect: rutina întoarce în registrele „H & L" adresa vectorului de alocare asociat discului selectat. Sistemul CP/M păstrează în memorie pentru fiecare unitate „activă", un vector de alocare. Acest vector poate fi folosit pentru a determina dimensiunea spațiului-disc rămas liber pe un volum (vezi comanda tranzitorie STAT).

OBSERVAȚIE: Informațiile cuprinse în vectorul de alocare asociat unei unități de disc care a fost desemnată ca R/O de către CP/M (în urma schimbării unui volum disc fără inițializarea sistemului CP/M sau fără o operație de inițializare stare sistem discuri" (rutina 13)) pot fi false.

RUTINA 28: Setare atribut R/O pentru o unitate de disc
(Write Protect Disk)

Intrări:

registrul C: 1CH

Efect: rutina desemnează temporar **discul selectat** ca **disc R/O.** Orice încercare de scriere pe acel disc, pînă la o inițializare sau reinițializare a sistemului CP/M sau pînă la o operație de „inițializare stare sistem discuri" (rutina 13), va produce mesajul:

BDOS ERR on d:R/O

RUTINA 29: Citire vector de unități R/O/
(Get Read/Only Vector)

Intrări:

registrul C: 1DH

Ieșiri:

registrele H & L: vectorul de unități R/O

Efect: rutina întoarce în registrele „H&L" un vector ce indică unitățile de disc care sînt desemnate ca R/O în acel moment. Bitul B0 din registrul „L" corespunde unității „A", iar bitul B7 din registrul „H" corespunde unității „P". O valoare logică „1" indică faptul că unitatea respectivă este R/O.

O unitate de disc devine R/O după un apel al rutinei 28 sau în urma schimbării volumului disc din acea unitate (sistemul CP/M, în acest caz, desemnează automat unitatea respectivă ca R/O).

RUTINA 30: Modificare atribute fişier
(Set File Attributes)

Intrări:

registrul C: 1EH
registrele D & E: adresa FCB

Ieşiri:

registrul A: octet de stare

Efect: rutina permite modificarea atributelor R/O şi SYS ale unui fişier specificat în FCB-ul a cărui adresă este dată în registrele „D & E". FCB-ul trebuie să conţină un specificator—individual de fişier. Noile atribute ale fişierului se specifică prin:

— bitul B7 din octetul 09 din FCB („1" reprezintă fişier protejat la scriere (R/O))
— bitul B7 din octetul 10 din FCB („1" reprezintă fişier invizibil (SYS)

Rutina caută în „director" o intrare care corespunde octeţilor 1—11 din FCB; comparaţia se face ignorînd valorile biţilor B7 din octeţii 1—11 din FCB şi din „director". Dacă o astfel de intrare este găsită rutina modifică corespunzător intrarea din „director" corespunzătoare. Rutina întoarce în registrul „A" o valoare 0—3 pentru cazul în care operaţia s-a desfăşurat corect sau o valoare egală cu 255 (0FFH) pentru cazul în care nu a fost găsită o astfel de intrare.

RUTINA 31: Citire adresa „bloc de parametri disc"
(Get ADDR (Disk Parms))

Intrări:

registrul C: 1FH

Ieşiri:

registrele H & L: adresa blocului de parametri ai discului

Efect: rutina întoarce în registrele „H & L" adresa „blocului de parametri ai discului", bloc care este rezident în BIOS. Această adresă poate fi folosită:

— pentru a extrage din zona respectivă informaţii privind parametrii discului (informaţii necesare pentru a fi afişate sau pentru a se realiza, pe baza lor, calcule)
— pentru a modifica, prin program, parametrii discului; **de obicei programele de aplicaţii nu folosesc rutine 31 în acest scop.**

RUTINA 32: Citire/Modificare număr utilizator curent (Set/Get User Code)

Intrări:

registrul C: 20H

registrul E: —0FFH (pentru citire număr utilizator curent)

— numărul utilizatorului curent (pentru modificarea numărului utilizatorului curent)

Ieşiri:

registrul A: numărul utilizatorului curent (dacă (E) = — 0FFH) sau nici o valoare (dacă (E) ≠ 0FFH)

Efect: rutina permite citirea numărului utilizatorului curent (dacă (E) =0FFH) şi întoarcerea acestui număr în registrul „A" sau modificarea numărului utilizatorului curent, în funcţie de valoarea curentă a registrului „E" (modulo 16). Numărul utilizatorului curent variază între 00H şi 0FH.

RUTINA 33: Citire directă (Read Random)

Intrări:

registrul C: 21H

registrele D&E: adresa FCB

Ieşiri:

registrul A: octet de stare

Efect: rutina este similară rutinei „citire secvenţială" (rutina 20) cu excepţia faptului că nu se citeşte din fişier înregistrarea de 128 de octeţi cu numărul specificat în octetul 32 din FCB, ci înregistrarea al cărei număr este dat în octeţii 33 şi 34 din FCB. Octeţii 33 şi 34 din FCB reprezintă o valoare pe 16 biţi cuprinsă între 0000H — 0FFFFH cu partea cea mai puţin semnificativă în octetul 33 şi cea mai semnificativă în octetul 34. Octetul 35 trebuie să fie 00H întrucît o valoare diferită de 00H indică o „depăşire" în afara sfîrşitului fişierului.

Citirea directă necesită în prealabil deschiderea „extensiei logice" cu numărul 0 a fişierului (prima „extensie logică" a fişierului), operaţie care se realizează prin rutina 15.

Dacă operaţia de citire directă s-a efectuat corect, atunci:

— registrul „A" va avea valoarea 00H

— înregistrarea citită din fişier se va găsi depusă în memorie la „adresa DMA"

— valorile octeţilor 12 (numărul „extensiei logice" curente) şi 32 (numărul înregistrării în cadrul „extensiei logice" curente) vor fi automat modificate în funcţie de numărul înregistrării citite (octeţii 33 şi 34)

— valoarea octetului 32 nu va fi incrementată cu 1 (ca în rutina 20)

OBSERVAŢII: 1. După o operaţie de „citire directă" pot fi folosite operaţii de „citire secvenţială"/„scriere secvenţială". Programatorul însă trebuie să ţină cont de faptul că prima operaţie de „citire secvenţială"/„scriere secvenţială" se va aplica asupra aceleiaşi înregistrări care s-a preluat prin „citire directă" (întrucît octetul 32 nu a fost incrementat cu 1). Se poate însă, printr-o „citire secvenţială" falsă incrementa, octetul 32 din FCB, astfel încît operaţiile de „citire/scriere secvenţială" următoare să se aplice asupra înregistrărilor care urmează celei preluate prin „citire directă".

2. Dacă operaţia de „citire directă" s-a aplicat asupra ultimei înregistrări dintr-o „extensie logică", nu se realizează **automat deschiderea „extensiei logice" următoare** (ca în rutina 20).

Dacă operaţia de „citire directă" nu s-a efectuat corect, atunci registrul „A" va conţine codul de eroare, respectiv:

— **01** citirea unei înregistrări nescrise

— **03** imposibilitate de închidere a „extensiei logice" curente (trebuie redeschisă sau recitită „extensia logică" numărul 0 a fişierului)

— **04** acces la o „extensie logică" a fişierului care nu a fost creată

— **06** octetul 35 este diferit de 00H (încercare de căutare în afara limitelor fizice ale fişierului)

În general, codurile de eroare diferite de 00H pot fi interpretate ca „lipsă de date".

RUTINA 34: Scriere directă
(Write Random)

Intrări:

registrul C: 22H
registrele D & E adresa FCB

Ieşiri:

registrul A: octet de stare

Efect: rutina este identică cu rutina de „citire directă" cu excepţia faptului că o înregistrare de 128 de octeţi aflată în memorie la „adresa DMA" este scrisă pe disc. Înregistrarea se va scrie în fişier în poziţia corespunzătoare numărului ei (octeţii 33 şi 34). Dacă acestei poziţii nu îi fusese alocat spaţiu, atunci rutina, înainte de scriere, realizează această alocare.

OBSERVAŢII: Dacă înregistrarea de scris nu are un număr astfel încît ea să fie prima înregistrare dintr-un bloc nealocat, atunci rutina va ocupa fictiv toate înregistrările, anterioare înregistrării de scris, din blocul respectiv de alocare.
Înregistrările ocupate fictiv (care sînt de fapt „găuri" în fişierul respectiv) **vor fi contabilizate în „contorul de înregistrări" din cadrul „extensiei logice" curente** (octet prezent

în intrarea de „director"), dar vor avea un conţinut aleator (vezi şi rutina 40).

În urma unei operaţii de „scriere directă" valorile octeţilor 12 şi 32 se modifică, dar octetul 32 nu se incrementează cu 1. Toate observaţiile referitoare la rutina 33 sînt valabile şi pentru rutina 34.

Dacă operaţia de „scriere directă" s-a efectuat corect, atunci registrul „A" va avea valoarea 00H; în caz contrar el va conţine codul de eroare. Codurile de eroare posibile sînt cele de la rutina 33 plus codul 05 care indică imposibilitatea scrierii datelor întrucît nu a mai fost spaţiu în „director" pentru crearea unei noi „extensii logice".

RUTINA 35: Determinare lungime fişier
(Compute File Size)

Intrări:

registrul C: 23H
registrele D & E: adresa FCB

Ieşiri:

lungimea virtuală a fişierului
(în octeţii 33, 34 şi 35 din FCB)

Efect: rutina necesită ca FCB-ul adresat prin registrele „D & E" să aibă 36 de octeţi şi să **conţină un specificator-individual de fişier.** Rutina caută în „director" informaţii privind fişierul specificat în FCB şi completează octeţii 33, 34 şi 35 cu o valoare egală cu numărul corespunzător primei înregistrări de pe disc care urmează **după sfîrşitul fizic al fişierului.** Astfel, octeţii 33, 34 şi 35 reprezintă „lungimea fişierului", lungime care poate fi:

— **lungime reală** (fizică) a fişierului (dacă fişierul a fost creat (scris) secvenţial)
— **lungime virtuală** a fişierului (dacă fişierul a fost creat în acces direct şi există „găuri" în alocarea fişierului)

Dacă octetul 35 are valoare egală cu 01H atunci, rezultă că fişierul conţine numărul maxim de înregistrări posibile (şi anume 65535).

Rutina poate fi folosită pentru a adăuga înregistrări într-un fişier. Astfel, prin apelul ei se determină numărul de ordine al primei înregistrări neocupate **de după sfîrşitul fizic al fişierului,** număr de ordine ce poate fi folosit în continuare de către o secvenţă de operaţii de „scriere directă".

RUTINA 36: Determinare număr înregistrare
(Set Random Record)

Intrări:

registrul C: 24H
registrele D & E: adresa FCB

Ieşiri:

numărul înregistrării

Efect: rutina întoarce în octeţii 33, 34 şi 35 din FCB numărul înregistrării curente dintr-un fişier care a fost citit/scris secvenţial. Rutina poate fi folosită astfel:

— pentru determinarea numărului de ordine al unor înregistrări dintr-un fişier, care conţin o anumită „cheie". În acest caz, fişierul se parcurge (în citire) secvenţial, se verifică dacă înregistrarea citită conţine „cheia" căutată şi dacă o conţine, atunci se apelează rutina 36 pentru a determina „numărul de ordine" al înregistrării respective. Acest număr de ordine se stochează şi apoi se continuă investigarea (secvenţială) a fişierului. La sfîrşitul prelucrării se va dispune de o listă a tuturor numerelor înregistrărilor care conţin „cheia" respectivă. Pe baza acestei liste, utilizatorul poate citi direct înregistrările care îl interesează.

— atunci cînd se doreşte trecerea de la accesul secvenţial într-un fişier la accesul direct. În acest caz, după ce un fişier a fost exploatat secvenţial pînă la un anumit punct, se apelează rutina 36 pentru a determina „numărul de ordine" al înregistrării curente. Pe baza acestui număr de ordine se pot realiza, în continuare, operaţii de citire/scriere directă (operaţii ce se aplică de la un anumit punct (selectat) din fişier în continuare).

RUTINA 37: Dezactivare discuri
(Reset Drive)

Intrări:

registrul C: **25H**

Ieşiri:

registrul A: **00H**

Efect: dezactivează unităţile de disc specificate în vectorul definit prin conţinutul registrelor „D & E" şi acordă acestor unităţi atributul R/W. Bitul B0 din registrul „E" corespunde unităţii „A" iar bitul B7 din registrul „D" corespunde unităţii „P". O valoare logică „1" în vectorul definit reprezintă o opţiune-utilizator de „dezactivare" a unităţii respective.

Rutina se foloseşte, de obicei, pentru a modifica atributul R/O, care a fost asociat unei unităţi de disc prin apelul rutinei 28.

OBSERVAŢIE: Discul selectat nu poate fi dezactivat prin această rutină ci numai printr-o rutină 13.

RUTINELE 38 şi 39: Aceste rutine nu au nici un efect în această versiune de sistem

RUTINA 40: Scriere directă cu umplere cu 0
(Write Random With Zero Fill)

Intrări:

registrul C: **28H**
registrele **D&E: adresa FCB**

Ieşiri:

registrul A: **octet de stare**

Efect: rutina este similară rutinei 34 (scriere directă) cu excepţia faptului că înainte de a se scrie o înregistrare, într-un bloc nealocat, acesta este automat umplut cu zerouri. Astfel, toate „găurile" dintr-un fişier creat în acces direct vor fi recunoscute prin conţinutul lor (zerouri).

ANEXA 1

Nr. rut (hex)	Denumire rutina	Intrări	Ieşiri
0	1	2	3
0	Reiniţializare sistem CP/M	C=00H	—
1	Citire caracter de la consolă	C=01H	A=caracter ASCII
2	Scriere caracter la consolă	C=02H E=caracter ASCII	—
3	Citire caracter de la dispozitivul „Reader" curent	C=03H	A=caracter ASCII
4	Scriere caracter la dispozitivul „Punch" curent	C=04H E=caracter ASCII	—
5	Scriere caracter la dispozitivul „List" curent	C=05H E=caracter ASCII	—
6	Citire/Scriere directă la consolă	C=06H E=0FFH =caracter ASCII	A=caracter ASCII sau =octet stare
7	Citire octet IOBYTE	C=07H	A=valoare octet IOBYTE
8	Modificare octet IOBYTE	C=08H E=valoare pentru octetul IOBYTE	—
9	Tipărire la consolă a unui şir de caractere	C=09H D & E=adresa şir	—
0A	Citire buffer consolă	C=0AH D & E=adresa buffer	—
0B	Citire stare consolă	C=0BH	A=stare consolă
0C	Citire versiune sistem	C=0CH	H & L=număr de versiune
0D	Iniţializare stare sistem discuri	C=0DH	—
0E	Selectare disc	C=0EH E=număr unitate selectată	—
0	Deschidere fişier	C=0FH D & E=adresa FCB	A=octet stare
10	Închidere fişier	C=10H D & E=adresa FCB	A=octet stare
11	Caută în „director" prima intrare	C=11H D & E=adresa FCB	A=octet stare
12	Caută în „director" următoarea intrare	C=12H	A=octet stare
13	Ştergere fişier	C=13H D & E=adresa FCB	A=octet stare
14	Citire secvenţială	C=14H D & E=adresa FCB	A=octet stare
15	Scriere secvenţială	C=15H D & E=adresa FCB	A=octet stare
16	Creare fişier	C&=16H DE= adresa FCB	A=octet stare

0	1	2	3
17	Schimbare nume fişier	C=17H D & E=adresa FCB	A=octet stare
18	Citire vector de unităţi-disc active	C=18H	H & L=vectorul de unităţi disc active
19	Citire număr disc selectat	C=19H	A=număr disc selectat
1A	Modificare „adresa DMA"	C=1AH D & E=adresa DMA	—
1B	Citire adresa vector de alocare	C=1BH	H & L=adresa vector de alocare
1C	Setare atribut R/O pentru o unitate de disc	C=1CH	—
1D	Citire vector de unităţi R/O	C=1DH	H & L=vectorul de unităţi R/O
1E	Modificare atribute fişier	C=1EH D & E=adresa FCB	A=octet stare
1F	Citire adresa „bloc de parametri disc"	C=1FH	H & L=adresa blocului de parametri disc
20	Citire/Modificare număr utilizator curent	C=20H E=0FFH număr utilizator curent	A=număr utilizator sau nimic
21	Citire directă	C=21H D & E=adresa FCB	A=octet stare
22	Scriere directă	C=22H D & E=adresa FCB	A=octet stare
23	Determinare lungime fişier	C=23H D & E=adresa FCB	lungime virtuală în octeţii 33, 34 şi 35 din FCB
24	Determinare număr înregistrare	C=24H D & E=adresa FCB	numărul înregistrare
25	Dezactivare discuri	C=25H	A=00H
26	NEIMPLEMENTATĂ		
27	NEIMPLEMENTATĂ		
28	Scriere directă cu umplere cu zero	C=28H	A=octet stare

Editat de I.T.C.I. Brașov

Bun de tipar: 25 V 88

Tiparul executat sub comanda nr. 1156
la Întreprinderea Poligrafică Brașov